2016-05-01

Rethinking the Systems Engineering Process in Light of Design Thinking

Giachetti, Ronald; Whitcomb, Clifford

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/53461

SYM-AM-16-024

PROCEEDINGS
OF THE
THIRTEENTH ANNUAL ACQUISITION RESEARCH SYMPOSIUM

WEDNESDAY SESSIONS
VOLUME I

Rethinking the Systems Engineering Process in Light of Design Thinking

Ronald Giachetti, Chair and Professor, NPS
Clifford Whitcomb, Professor, NPS

Published April 30, 2016

Prepared for the Naval Postgraduate School, Monterey, CA 93943.

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL

The research presented in this report was supported by the Acquisition Research Program of the Graduate School of Business & Public Policy at the Naval Postgraduate School.

To request defense acquisition research, to become a research sponsor, or to print additional copies of reports, please contact any of the staff listed on the Acquisition Research Program website (www.acquisitionresearch.net).

Panel 3. Systems Engineering: New Thinking for a New Age

Wednesday, May 4, 2016	
11:15 a.m. – 12:45 p.m.	**Chair: John D. Burrow,** Deputy Assistant Secretary of the Navy for Research, Development, Test, & Evaluation (DASN RDT&E) ***Rethinking the Systems Engineering Process in Light of Design Thinking*** Ronald Giachetti, Chair and Professor, NPS Clifford Whitcomb, Professor, NPS ***Content Analysis in Systems Engineering Acquisition Activities*** Karen Holness, Assistant Professor, NPS ***Update on the Department of the Navy Systems Engineering Career Competency Model*** Clifford Whitcomb, Systems Engineering Professor, NPS Corina White, Systems Engineering Research Associate, NPS Rabia Khan, Research Associate, NPS Dana Grambow, Research Psychologist, OPM Jessica Delgado, Technical Workforce Strategy Lead, DASN (RDT&E) José Vélez, Technical Workforce Lead, DASN (RDT&E)

Rethinking the Systems Engineering Process in Light of Design Thinking

Ronald E. Giachetti—is the Chair and Professor of the Systems Engineering Department at the Naval Postgraduate School (NPS) in Monterey, CA. He teaches and conducts research in the design of enterprise systems, systems modeling, and system architecture. He has published over 50 technical articles on these topics, including a textbook, *Design of Enterprise Systems: Theory, Methods, and Architecture*. Prior to joining NPS, he was at Florida International University in Miami, FL. He earned his BS in mechanical engineering from Rensselaer Polytechnic Institute, MS in manufacturing engineering from Polytechnic University (at NYU), and PhD in industrial engineering from North Carolina State University. [regiache@nps.edu]

Clifford Whitcomb—is a Professor and former Chair of the Systems Engineering Department at the Naval Postgraduate School (NPS) in Monterey, CA. He teaches and conducts research in model-based systems engineering and leadership, communication, and interpersonal skills development for engineers. He is the co-author of *Effective Interpersonal and Team Communication Skills for Engineers*. He is a principal investigator for research projects from the U.S. Navy Office of Naval Research, Office of the Joint Staff, Office of the Secretary of the Navy, and several naval system commands and naval warfare centers. He is an INCOSE Fellow. He is a retired naval officer, having served 23 years as a submarine warfare officer and Engineering Duty Officer. He earned his BS in nuclear engineering from the University of Washington in Seattle, WA, in 1984; MS degrees in naval engineering and electrical engineering and computer science from MIT in 1992; and PhD in mechanical engineering from the University of Maryland in College Park, MD, in 1998. [cawhitco@nps.edu]

Abstract

The systems engineering process to design and develop new systems is based on a technical rationalization of the design process. This paper contrasts the technical rational approach with the design thinking approach, which describes the principles and methods based on how experienced designers approach design problems. We assert the structure of the design problem changes during development, and one contributor to the challenges that defense programs face in meeting budget, schedule, and performance requirements is the mismatch between the nature of the design problem and the engineering approach. Our position is a variant of contingency theory, contending there is no single best way to approach a problem, and an approach effective in one situation may not be effective in another. This paper reviews the technical rational and design thinking perspectives. The paper then examines the systems engineering process in light of design thinking principles and methods, and the paper makes recommendations to partition development into architecting and engineering, increase the variety and frequency of prototyping, explicitly show iteration in process models, and practice delayed commitment.

Introduction

The defense acquisition system implements systems engineering through standards, codification of policies and procedures, and extensive documentation. The systems engineering vee is the process model and serves as the de facto standard process model for Department of Defense (DoD) programs. The vee process model is a top-down approach of analyzing stakeholder needs to arrive at technical system requirements and finally a system design. The top-down approach is evidence in the extensive decomposition from the system-level design to subsystem design and component design. The vee model then shows synthesis by building and integrating the system from a bottom-up perspective. This is followed by component level, subsystem level, and finally system-level test and evaluation. The vee model makes feedback explicit in verification and validation information flows from test and evaluation to the analysis and design activities.

The systems engineering vee model adheres to the technical rational perspective. In this paper, we review the technical rational design approach and the assumptions underlying its methods. We then introduce design thinking and its assumptions. The technical rational design approach and the design thinking approach start with different worldviews and lead to two very different design approaches. We then analyze the systems engineering process in order to make recommendations to improve the process. We make recommendations and draw final conclusions.

Technical Rational Design

The Technical Rational Design approach is a structured approach to design based on a problem-solving perspective in which the designer's task is to solve a design problem. Simon (1996) was among the first to present the problem-solving perspective of design, which separates design into a problem formulation phase and problem solution phase. Simon and the artificial research community at the time sought computer algorithms to do the design process. The technical rational design approach assumes a positivist perspective that a single objective truth exists and can be observed and discovered through scientific methods (Neuman, 2005).

Pahl and Beitz (2013) wrote an influential German text defining a systematic approach to engineering design, which illustrates the assumptions and perspective of technical rational design. They partition the design process into four phases of clarifying the task, conceptual design, embodiment design, and detail design. The design process starts with the definition of requirements followed by successful refinement of a design concept through the last three phases. Each step of the way, the designer is making rational decisions in a pre-determined manner to arrive at the final design.

The technical rational design approach makes two key and interrelated assumptions. First, technical rational design approach assumes problem formulation can be separated from problem solution. We see evidence of this mindset in many texts with the advice to separate the "what" described by the functional architecture from the "how" described by the physical architecture (see Blanchard & Fabrycky, 1990). Second, the technical rational design approach assumes we can know and present the stakeholder objectives and system requirements without embarking on any design activities. The designer would then be able to search the design space to determine the set of Pareto optimal designs.

Given these two assumptions, design can progress in an orderly fashion through each step with minimal feedback and iteration. Moreover, adopting these assumptions makes the design problem amenable to formulation as a mathematical problem, which can then be subjected to algorithms to find the best designs. Here we formulate the design problem.

Design variables are the controllable dimensions, characteristics, and attributes of a system design specification. Initially, the value for each design variable is unknown, and through the process of design, the designer will specify values for the design variable until all design variables are specified. Let d_i denote the i^{th} design variable which can take any value in Δ_i, in other words $d_i \in \Delta_i$. The set Δ_i can be the set of integers, real numbers, or discrete options available for that design parameter (e.g., if d_i is the design parameter for battery type, then the domain Δ_i = {lithium-ion, nickel-cadmium, lead-acid}). If there are n design parameters, then the *design space* is an n-dimensional hyperspace that contains all the possible designs. It is defined by the Cartesian product

$$DS = \Delta_1 \times \Delta_2 \times \ldots \times \Delta_n$$

A design denoted by $\mathbf{D^k}$ is a vector of length n that specifies a value for each of the design variables, i.e., $\mathbf{D^k} = \left(d_1^k, \ldots, d_n^k\right)$. The superscript denotes the k^{th} design and distinguishes between the many designs in a design space. Every point in the design space is a design. However, not every design in *DS* will satisfy stakeholder requirements or even be technically feasible.

Requirements either describe function relationships between multiple design variables or requirements place restrictions on the admissible values of a design variable. The j^{th} function requirement is given by

$$r_j\left(\mathbf{D^k}\right) \leq 0 \quad j = 1 \ldots m$$

and requirement restrictions are expressed by lower limits Δ_i^{ll} and upper limits Δ_i^{ul} on the admission values as $\Delta_i^{ll} \leq d_i \leq \Delta_i^{ul}$.

The j^{th} system requirement partitions the design space into a region that satisfies the requirement, DS_j^S and a region that does not satisfy the requirement, DS_j^N. A design $\mathbf{D^k}$ satisfies a system requirement if it is in the satisfactory region of the requirement defined by $DS_j^S \subseteq DS$. The intersection of all m requirements defines the *satisfactory region* within which each design satisfies all the system requirements, and is given by

$$DS^S = \bigcap_{j=1}^{m} DS_j^S .$$

A design team will seek the best design, in other words the design that delivers the most value to the stakeholders, within the satisfactory region. Almost all designs will have multiple objectives from which stakeholders derive value. The value of a design with respect to a single objective is given by a *value function*. Value is a function of the design parameters and noise parameters. The value of the k^{th} design with respect to the l^{th} objective is given by the *value function*

$$V_l^k = f\left(d_1^k, \ldots, d_n^k, n_1, \ldots, n_p\right).$$

The set of noise parameters, denoted by $n_1, \ldots, n_p$, represents uncontrollable influences on performance such as environmental factors.

The vector $\mathbf{V^k} = \left(V_1^k, \ldots, V_n^k\right)$ denotes the values of the k^{th} design across all objectives. A design with a value of $\mathbf{V}^a = \left(V_1^a, \ldots, V_n^a\right)$ is said to dominate a design with a value of $\mathbf{V}^b = \left(V_1^b, \ldots, V_n^b\right)$ if and only if $\mathbf{V}^a$ is partially less than $\mathbf{V}^b$, which is when $\forall l \in L\, V_l^a \geq V_l^b \wedge \exists l \in L, V_l^a > V_l^b$.

The set of dominate designs is called the Pareto frontier. We speak of designers trading off objectives, and they would do this between designs in the Pareto frontier.

In summary, the design problem is formulated as finding the design(s) that maximize value while satisfying all the system requirements. It is expressed by the optimization model

$$\begin{aligned} \underset{\mathbf{D^k}}{\arg\max} \qquad & \mathbf{V^k} = \left(V_1^k, \ldots, V_n^k\right) \\ & r_j\left(\mathbf{D^k}\right) \le 0 \qquad j = 1 \ldots m \\ & \Delta_i^{ll} \le d_i \le \Delta_i^{ul} \qquad i = 1 \ldots n \end{aligned}$$

The design optimization model is possible in technical rational design because the problem structure is assumed to be well-defined, it is assumed we can express value mathematically, and it is assumed we can express all requirements as mathematical functions. The design problem then becomes a matter of searching the design space to find the Pareto optimal designs.

The concepts and assumptions of the technical rational design approach form the basis upon which systems engineering process models (Blanchard & Fabrycky, 1990) and the majority of engineering design education (Dym et al., 2005). The waterfall model was an early example, largely developed in reaction to the poor experience of development software without any process.

There are many benefits to the technical rational design approach embodied by these methods. The systemization of design leads to manageable projects and the ability to define milestones and deliverables, and it standardizes the process which facilitates communication and makes the process repeatable. These benefits have enhanced government's and industry's ability to design and develop complex weapon systems.

Design Thinking

Design thinking is a term to describe the creative thinking process exhibited by designers and now used in many non-traditional design domains such as strategy formulation, business, and social sciences (Brown, 2008; Plattner et al., 2010). It has also influenced the Navy, as seen in ADM Richardson's eight-page "A Design for Maintaining Maritime Superiority" strategic document.

Dorst (2010) differentiates design thinking from other thought processes through the logical process of abduction, whereby we know the end value we want and have to discover the means to achieve it. The study and conceptualization of design thinking is conducted primarily according to an interpretivism approach after Schön's (1983) reflection-in-action research, in which he examined how professionals actually work. Interpretivism accepts multiple different realities based on the observer's perspective. It is in contrast to the positivist's claim that there is a single objective reality and we can only acquire knowledge through the scientific method, which is the technical rational approach (Neuman, 2005).

A process for design thinking identifies five activities (after Stanford University Institute of Design, 2016):

1. Empathize—Understand what the stakeholders desire through open-ended questions and related techniques to better understand the problem from many different perspectives.
2. Define—Combine and synthesize all the acquired information and perspectives to arrive at a group consensus on the problem structure.
3. Ideate—Generate ideas in a typical brainstorming fashion with the goal to generate as many ideas as possible.

4. Prototype—Create a mock-up of the design solution and use it for evaluation.
5. Test—Test the prototype, preferably with stakeholders and end-users.

Completion of a single iteration leads to a greater understanding of the problem as well as a potential design solution. Design thinking is based on the observation that designers work simultaneously on both problem structuring and problem solving (Dorst & Cross, 2001). Problem structuring involves the discovery of needs, requirements, and feasibility so that the designer can understand the problem. Problem structuring is achieved partially by proposing solutions because having a solution provides something concrete for stakeholders to react to and better understand their needs.

Design thinking is also referred to as human-centric design because of the importance placed on empathizing with the human users (Patnaik, 2009). During the empathize step, designers frame and re-frame the problem by adopting the user's perspective to arrive at different problem structures. Framing the problem from multiple perspectives implies the imposition of an interpretation of the problem, and each interpretation allows for additional insights and potentially different and more fruitful solutions (Paton & Dorst, 2011).

Unlike technical rational design, design thinking seeks to preserve ambiguity as long as possible because too quickly converging on a solution is seen to stifle creativity. Design thinking also promotes the early and frequent creation of prototypes to serve multiple purposes from problem understanding, solution evaluation, and communication.

Analysis and Recommendations

This section is organized according to the main recommendations on how design thinking can be incorporated into the systems engineering process.

Architecting vs. Engineering

The design problem changes in character from an ill-structured problem in the early phases to a well-structured problem in the later phases. Consequently, it makes sense to approach the different design problems differently. The concept of tailoring is based on contingency theory, which claims the best approach depends on the fit between the process and contextual factors (Drazin & Van de Ven, 1985). In the systems engineering process, a major contextual factor is the nature of the design problem: ill-structured or well-structured.

DoD Instruction (DoDI) 5000.02 (*Operation of the Defense Acquisition System*) allows for tailoring and says, "The structure of a DoD acquisition program and the procedures used should be tailored as much as possible to the characteristics of the product being acquired, and to the totality of circumstances associated with the program including operational urgency and risk factors." The instruction provides four baseline acquisition models to serve as starting points for tailoring. What is lacking in the systems engineering community is guidance on how to make the tailoring decisions.

The design process should be partitioned between two distinct phases of architecture design and system design. The architecture phase should be managed according to a design thinking approach, and the system design phase according to the technical rational design approach. Architecting is the activity comprising the generation, evaluation, and selection of alternative solutions. The architect works in both the problem space and the design space. Understanding the problem and conceiving of a design solution are directly related to each other. Consequently, architects iterate between problem structuring and problem solving and in the process they reveal new understandings of the problem space and the solution space.

The output of the architecture design phase is a system architecture defining the structure of the system in terms of the design variables, set of system technical requirements, and the measures of effectiveness, which in the DoD define value. Consequently, we have a well-defined problem amenable to the technical rational design approach. Designers would search the design space using algorithms and computational tools when available and appropriate to find the set of Pareto optimal design solutions.

We note the systems engineering community has been moving to this dichotomy between system architecting and system engineering, as evidenced by the earliest book on system architecture (Rechtin & Maier, 2010), to more recent works and emphasis (Dickerson & Mavris, 2009).

Requirements

Both technical rational design and design thinking suggests we need to think of systems requirements as being of two types: value statements and technical system requirements. Value statements express what stakeholders value in a system, can be measured on a continuous scale, and are negotiable. Requirements are the constraints a system must have and are non-negotiable. In the design optimization model, the value statements are part of the objective function and the requirements define the edges of the design space. When we state stakeholder value as a requirement rather than a value statement, we shackle the hands of our designers by unnecessarily restricting the design space. The value statements more closely match attainment of value as defined by stakeholders. Barry Boehm came to a similar conclusion and suggested we need to modify our terminology in order to effect the cultural change within the acquisition and systems engineering communities (Mavor & Pew, 2007).

Since the set of requirements define the edges of the design space, it is easily shown that adding requirements makes the design space smaller or at best the same size. If the design space is made smaller, then it is possible good designs are excluded. Given this insight, it is important to keep to a minimum the number of technical system requirements because they limit, perhaps unnecessarily in some cases, the design space.

Prototyping

Prototyping during the early architecting phase is as important as during the later phases (Kimbell, 2011). It seems many programs illogically think a prototype is an almost fully-functional copy of the intended system. Prototyping in the design thinking community is much more inclusive. Prototyping during the architecting phase is important for reasons of discovery, developing a deeper understanding of stakeholder value, communication, and to support problem structuring. A prototype as discussed by the design thinking community is any physical model that stakeholders and the designers can interact with. Design thinking promotes the building and usage of many low fidelity prototypes to aid the designers during problem structuring. An overemphasis by many programs on high fidelity prototypes with much of the functionality of the expected production system is counterproductive because they overlook the value of prototyping in the early architecting phase. Programs need to expand their prototyping capability in terms of both the diversity and fidelity of prototypes.

Incremental and Iterative

Design thinking research has demonstrably revealed that higher performing designers iterate between problem structuring and problem solving (Dorst & Cross, 2011). Top-down, sequential process models such as the vee model do not show this important aspect of system design and development. Moreover, the systems engineering vee and the Joint Capability Integrated Development (JCIDS) process suggest it is possible for the government to generate a solution agnostic specification of capability needs and system requirements. Design thinking says such a separation is not possible. In fact, designers need to think about solutions in order to better understand needs and system requirements. The systems engineering models should incorporate documentation to stress the importance of both incremental and iterative development. Larman and Basili (2003) discuss the history of incremental and iterative development and why within the software domain these methods are usually superior to sequential and document-intensive methods.

The number of iterations in iterative approaches is limited by either time or budget. Consequently, it is impossible to exhaustively search the entire design space before running out of time or money. All iterative approaches are local searches confined by the starting point and consequently, if you have a poor starting point, you will likely finish at an inferior design. One strategy is the multi-start whereby instead of using a single starting design to iterate upon, the designers consider multiple alternative designs preferably representative of the entire design space. Indeed, a GAO (2009) report analyzed 32 major defense programs that started after the year 2003. The GAO found the programs with a broad scope of alternatives had lower cost and schedule growth than programs with a narrow scope of alternatives. Each alternative is essentially a starting design for a multi-start strategy to explore the design space. A broader AoA is more likely to fully explore the design space and lead to better program outcomes. A narrow AoA is less likely to fully explore the design space; hence the problems.

Deferment and Delayed Commitment

The architecting phase is characterized by high uncertainty, yet it is well established that early design decisions can have an enormous impact on committed cost (Blanchard & Fabrycky, 1990). Deferring decisions until more information can be gained is a good strategy (Loch & Terwiesch, 2005). Set-based design, based upon American understanding of Toyota's design process, is when instead of iterating from a starting design, a set of designs is propagated and progressively pruned until a final design is found (Sobek et al., 1999). Set-based design is one approach to tackling the mismatch between the amount of information available and the timing of decisions. It delays decisions until more information is available. This is a form of progression refinement since as the development process progresses, the uncertainty (measured as the size of the set) is gradually decreased until a precise value is arrived at. Giachetti et al. (1999) did something similar with fuzzy sets; Finch and Ward (1995) with intervals; HP with delayed differentiation; and Boehm and Lane (2007) with delayed commitment. More recently, the set-based approach has been applied to naval ship design (Singer et al., 2009; Mebane et al., 2011).

Conclusions

Design thinking starts out with a very different worldview from the technical rational design approach. While technical rational design is based on a positivist perspective of knowledge, design thinking is based on an interpretative perspective. The result is very different assumptions about how to conduct design, and consequently very different approaches. Using contingency theory, we propose to partition the system design and development process to achieve a better match between the problem space and the solution approach. Broadly, this means separating design and development into two phases of architecting and engineering design. The architecting phase is guided primarily by the design thinking perspective, and the engineering design phase is guided primarily by the technical rational design perspective. Additionally, we make recommendations for adoption of a broader set of prototyping capabilities, rethinking many requirements as value statements, and for greater recognition of iteration and incremental development in the systems engineering process model. The Systems Engineering Department at the Naval

Postgraduate School (NPS) is working towards educating the younger cohort of naval engineers in design thinking and how it can be beneficially incorporated into the systems engineering process.

References

Blanchard, B. S., Fabrycky, W. J., & Fabrycky, W. J. (1990). *Systems engineering and analysis* (Vol. 4). New Jersey: Prentice Hall.

Boehm, B., & Lane, J. A. (2007). Using the incremental commitment model to integrate system acquisition, systems engineering, and software engineering. *CrossTalk, 19*(10), 4–9.

Brown, T. (2008). Design thinking. *Harvard Business Review, 86*(6), 84.

Dickerson, C., & Mavris, D. N. (2009). *Architecture and principles of systems engineering*. CRC Press.

Dorst, K., (2010). The nature of design thinking. In *DTRSI Interpreting Design Thinking: Design Thinking Research Symposium* (pp. 131–139).

Dorst, K., & Cross, N., (2001). Creativity in the design process: Co-evolution of problem and solution. *Design Studies, 22*(5), 425–437.

Drazin, R., & Van de Ven, A. H. (1985). Alternative forms of fit in contingency theory. *Administrative Science Quarterly, 30*(4), 514–539.

Dym, C. L., Agogino, A. M., Eris, O., Frey, D. D., & Leifer, L. J. (2005). Engineering design thinking, teaching, and learning. *Journal of Engineering Education, 94*(1), 103–120.

Finch, W. W., & Ward, A. C. (1995). Generalized set-propagation operations over relations of more than three variables. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing, 9*(03), 231–242.

GAO. (2009). *Many analyses of alternatives have not provided a robust assessment of weapon system options* (GAO-09-665). Washington, DC: Author.

Giachetti, R. E., Young R. E., Roggatz, A., Eversheim W., & Perrone G. (1997). A methodology for the reduction of imprecision in the engineering design process. *European Journal of Operations Research, 100*(2), 277–292.

Kimbell, L. (2011). Rethinking design thinking: Part I. Design and Culture, 3(3), 285–306.

Larman, C., & Basili, V. R. (2003). Iterative and incremental development: A brief history. *Computer*, (6), 47–56.

Loch, C. H., & Terwiesch, C. (2005). Rush and be wrong or wait and be late? A model of information in collaborative processes. *Production and Operations Management, 14*(3), 331–343.

Mavor, A. S., & Pew, R. W. (Eds.). (2007). *Human-system integration in the system development process: A new look*. National Academies Press.

Mebane, W. L., Carlson, C. M., Dowd, C., Singer, D. J., & Buckley, M. E. (2011). Set-based design and the ship to shore connector. *Naval Engineers Journal, 123*(3), 79–92.

Neuman, W. L. (2005). *Social research methods: Quantitative and qualitative approaches* (Vol. 13, pp. 26–28). Boston, MA: Allyn and Bacon.

Pahl, G., & Beitz, W. (2013). *Konstruktionslehre: Methoden und Anwendung*. Springer-Verlag.

Patnaik, D. (2009). *Wired to care: How companies prosper when they create widespread empathy.* Upper Saddle River, NJ: Pearson Education.

Paton, B., & Dorst, K. (2011). Briefing and reframing: A situated practice. *Design Studies, 32*(6), 573–587.

Plattner, H., Meinel, C., & Leifer, L. (2010). *Design thinking: Understand-improve-apply.* Springer Science.

Rechtin, E., & Maier, M. W. (2010). *The art of systems architecting*. CRC Press.

Schön, D. (1983). *The reflective practitioner: How professionals think in action*. New York, NY: Basic Books.

Simon, H. A. (1996). *The sciences of the artificial*. Cambridge, MA: MIT Press.

Singer, D. J., Doerry, N., & Buckley, M. E. (2009). What is set-based design? *Naval Engineers Journal, 121*(4), 31–43.

Sobek, D. K., II, Ward, A. C., & Liker, J. K. (1999). Toyota's principles of set-based concurrent engineering. *MIT Sloan Management Review, 40*(2), 67.

Stanford University Institute of Design. (2016). Retrieved from http://dschool.stanford.edu/dgift/

Ullman, D. G. (1992). *The mechanical design process* (Vol. 2). New York, NY: McGraw-Hill.

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL
555 DYER ROAD, INGERSOLL HALL
MONTEREY, CA 93943

www.acquisitionresearch.net